



CASCADING STYLE SHEETS

LAYOUTS DE PÁGINA

OBJETIVOS DE APRENDIZAGEM

- Conhecer abordagens e estratégias para criação layouts de página utilizando CSS

AGENDA

- Estratégias de *layout* de página
- Técnicas de *layout* de página
- *Layouts* multi-coluna
- *Layouts* posicionados

ESTRATÉGIAS

- Considerar o fato de o usuário utilizar telas de diferentes tamanhos
- O usuário pode ainda a possibilidade de redimensionamento da janela
- As principais estratégias de *layout* para solucionar essa questão recaem nas categorias:
 - *Layouts* fixos (*Fixed*)
 - Fluidos (*Fluid* ou *Liquid*)
 - Elásticos (*Elastic*)
 - Híbridos (*Hybrid*)

LAYOUT FIXO

- Possuem **largura pré-determinada** pelo *designer*
- Controle total sobre os elementos da página (área de impressão, comprimentos de linha, posicionamento, etc)
- Voltado para usuários que utilizam basicamente o *desktop*

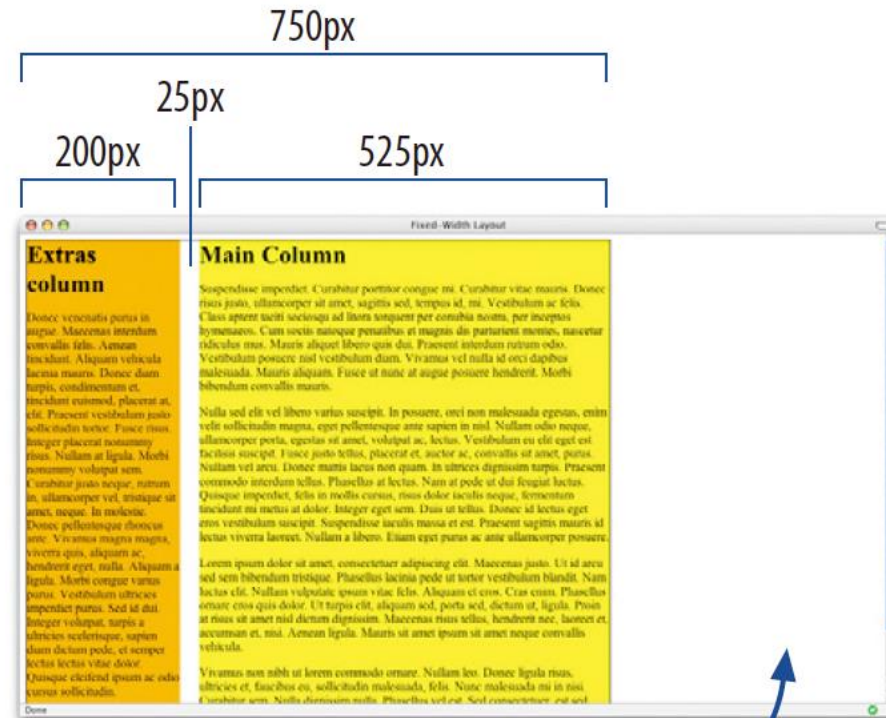
LAYOUT FIXO

- Pontos a serem observados:
 - Resolução da tela
 - Resoluções menores tornam o *site* compatível com mais usuários, por exemplo 960 pixels (1024 x 768)
 - Posicionamento do *layout*
 - Esquerda, deixando espaço excedente a direita
 - Centralizado

```
#wrapper {width: 750px;
position: absolute;
margin-left: auto;
margin-right: auto;
border: 1px solid black;
padding: 0px;}
```

```
#extras {position: absolute;
top: 0px;
left: 0px;
width: 200px;
background: orange; }
```

```
#main {margin-left: 225px;
background-color: yellow;}
```



LAYOUT FIXO

- Como fazer:
 - Especificando `widths` em **pixels**
 - Criar uma `div` para o site inteiro (content, container, wrapper, page, etc) com uma largura específica
 - Especificar a largura das colunas, além de `margin` e `padding`

LAYOUT FIXO

vantagens

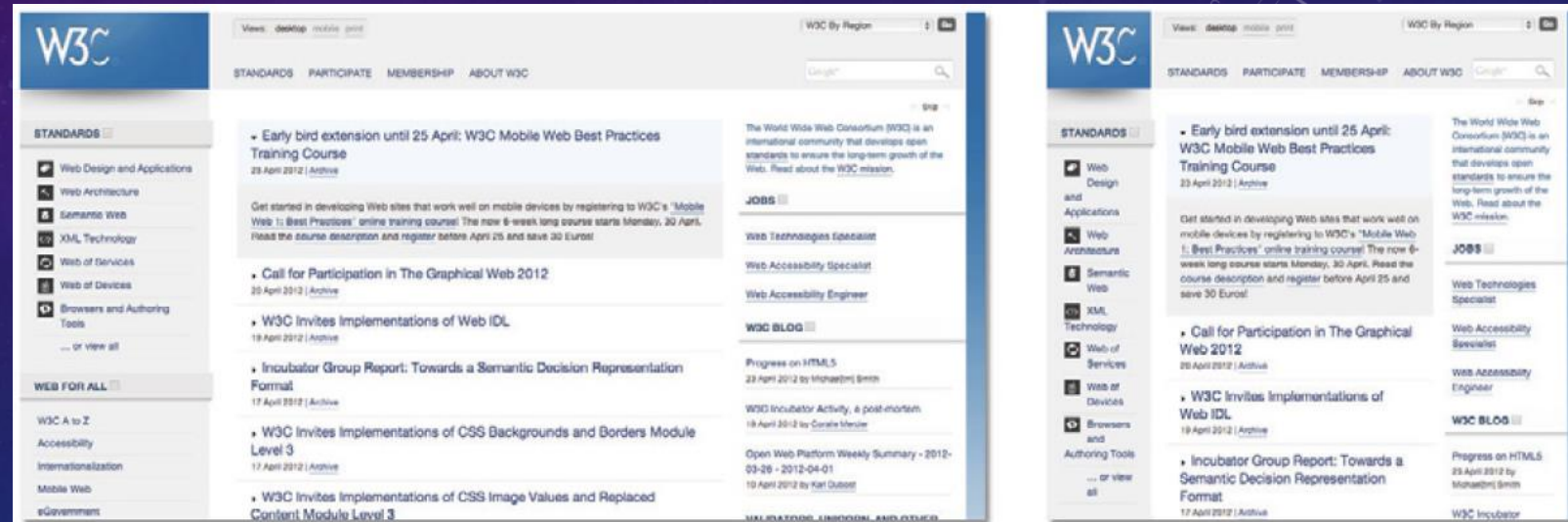
- Previsível, com grande controle sobre a visualização
- **Fácil** de implementar

desvantagens

- Conteúdo a direita será escondido caso a janela do navegador seja menor que a largura especificada
- Em telas maiores haverá grandes espaços laterais

FLUID LAYOUT

- A principal característica é o **redimensionamento dos elementos da página de acordo com a janela** do navegador
- O comportamento é similar ao fluxo normal
- O texto/conteúdo se ajusta conforme necessidade (largura da janela)
- Base para *designs* responsivos
- <https://www.w3.org/>



FLUID LAYOUTS

vantagens


- Aproveita as vantagens do ambiente que está inserido (*desktop, mobile, etc*)
- Evita áreas vazias (conteúdo sempre preenche a janela)
- Tamanhos de janela controlados pelo usuário em telas maiores
- Sem barras de rolagem

desvantagens

- Comprimentos de linhas podem se tornar difíceis de ler em telas grandes
- Elementos podem ser exibidos de forma inesperada (muito estreitos ou muito largos)
- Mais cálculos envolvidos na criação

COMO CRIAR FL?

- Especificar width em %
- O valor default de width é auto, então uma opção seria omitir esse atributo

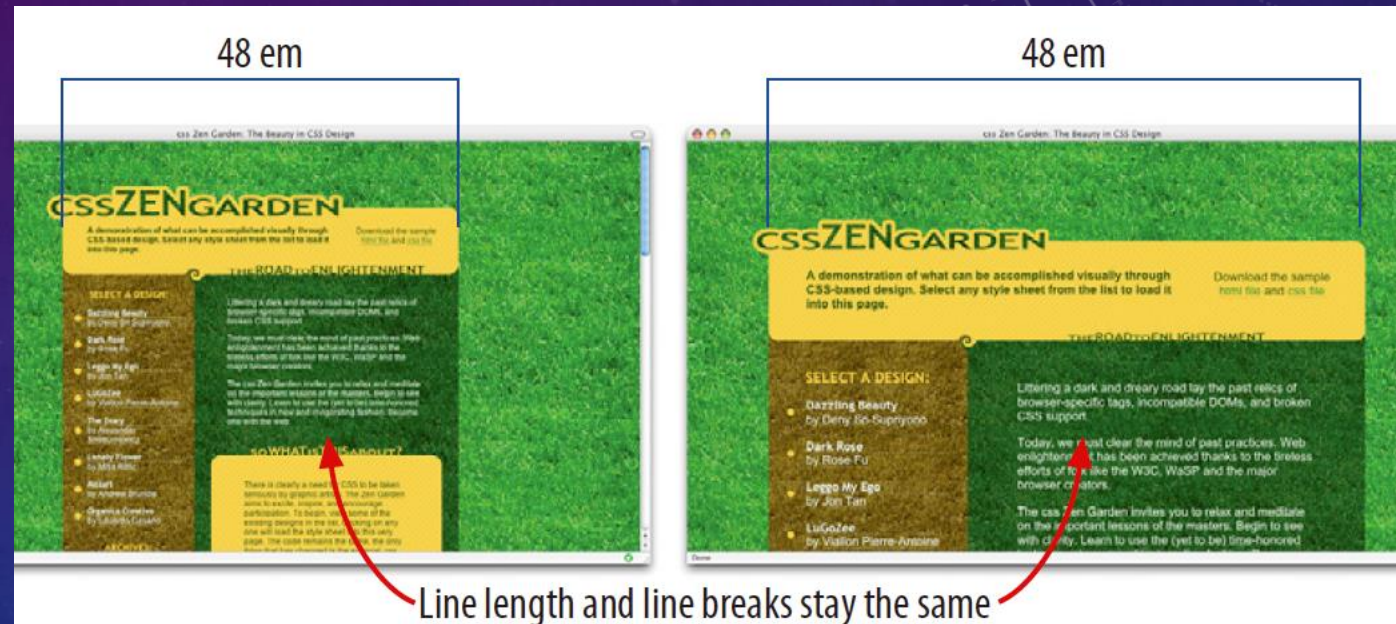


The diagram illustrates a liquid layout with two columns. The 'Main Column' is labeled with a width of 70% and contains placeholder text. The 'Extras column' is labeled with a width of 25% and a margin-right of 5% from the main column. The browser window title is 'Liquid Layout'.

```
div#main {  
  width: 70%;  
  margin-right: 5%;  
  float: left;  
  background: yellow;  
}  
  
div#extras {  
  width: 25%;  
  float: left;  
  background: orange;  
}
```


ELASTIC LAYOUTS

- Combinam **texto ajustável com comprimentos de linha previsíveis**
- Caso o usuário amplie a largura da janela o box será expandido e vice-versa
- Cria uma associação entre as **proporções** da página e o conteúdo textual
- <http://www.csszengarden.com/063/>



ELASTIC LAYOUTS

vantagens

- *Layout* proporcional ao tamanho do texto
- Maior controle do comprimento das linhas em comparação com os layouts fixos e líquidos

desvantagens

- Outros conteúdos não textuais (imagens, vídeos, etc) não ajustam-se ao restante dos elementos
- Pode exceder o tamanho da janela do navegador
- Pouca utilidade em sites responsivos (pode se tornar pouco legível)
- Mais complexo de criar

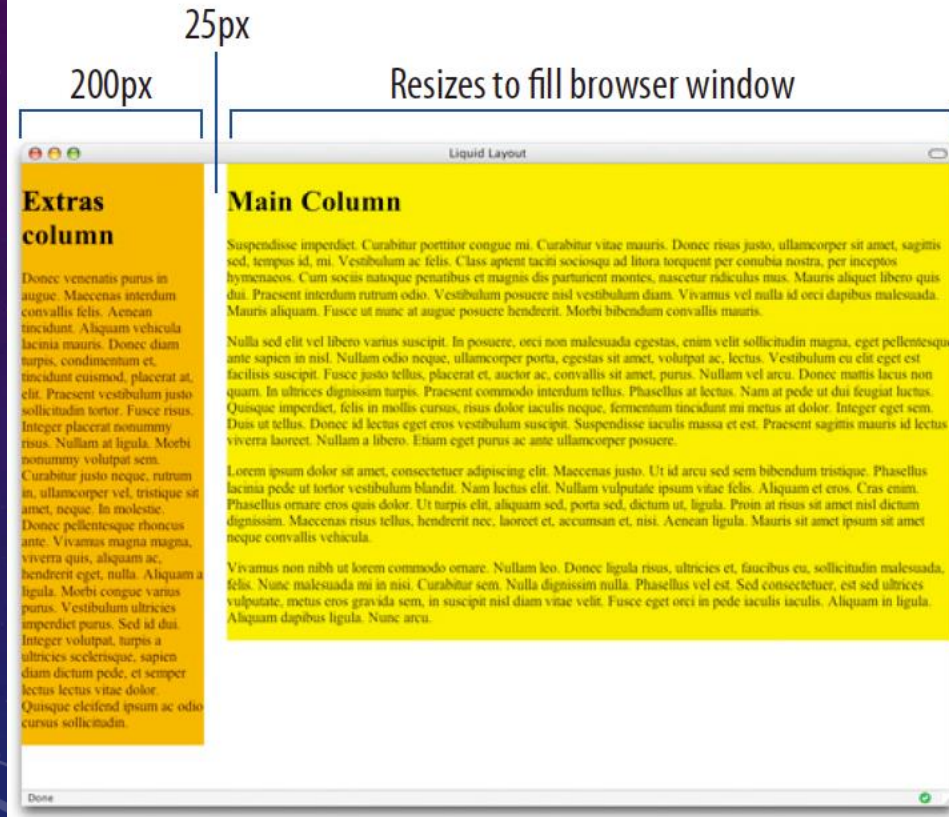
COMO CRIAR EL?

- Utilizar unidades de medida `em`
- Os elementos devem ser especificados relativo ao tamanho da fonte
- Exemplo, para a fonte 16 em um box com `width: 40em` teria 640 pixels ($40\text{em} \times 16\text{px/em}$)
 - Caso a fonte passe para 20, o box aumenta para 800 pixels

LAYOUT HÍBRIDO

- Utilizam a combinação de medidas em `px`, `em` e `%`
- *Sidebars* e *banners*, normalmente possuem tamanhos fixos
- Conteúdos, normalmente podem ser ajustados com o tamanho
- LH são os mais complexos, devido a mescla de unidades de medida diferentes

LAYOUT HÍBRIDO



fixed-width and auto sized

```
div#main {  
  width: auto;  
  position: absolute;  
  top: 0;  
  left: 225px;  
  background: yellow; }
```

```
div#extras {  
  width: 200px;  
  position: absolute;  
  top: 0;  
  left: 0;  
  background: orange; }
```

TÉCNICAS DE LAYOUT

1. *Floating layouts*

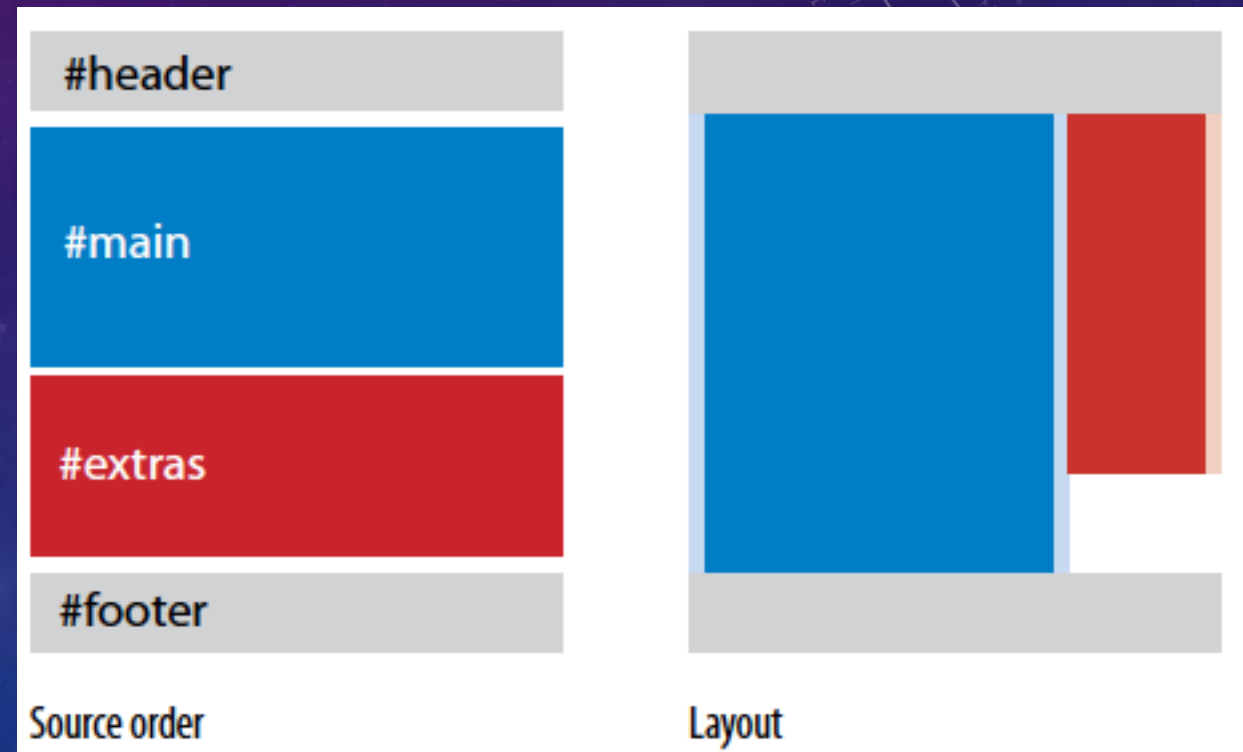
- a. *Layout Fluid* de 2 columnas
- b. *Layout Fixo* de 2 columnas
- c. *Layout Fluid* de 3 columnas

2. *Positioned layouts*

- a. *Fluid* de 3 columnas
- b. *Fixo* de 3 columnas

FLUID DUAS COLUNAS

- Fixar `widths` das duas colunas e `float` a esquerda das duas colunas
- Usar `clear` para manter o rodapé na parte inferior



- `div header` com fluxo normal
- As `divs main` e `extra` com `float: left`
- `Div main` com `margin left` e `right` especificada
- `Div extras` com `margin left` especificada
- `Div footer` com `clear: left` para ficar abaixo das `divs` anteriores
- Acrescente cores e conteúdo ao código. Como fica a exibição?
- Como usar `max-width` e `min-width` na `div main`?

The markup

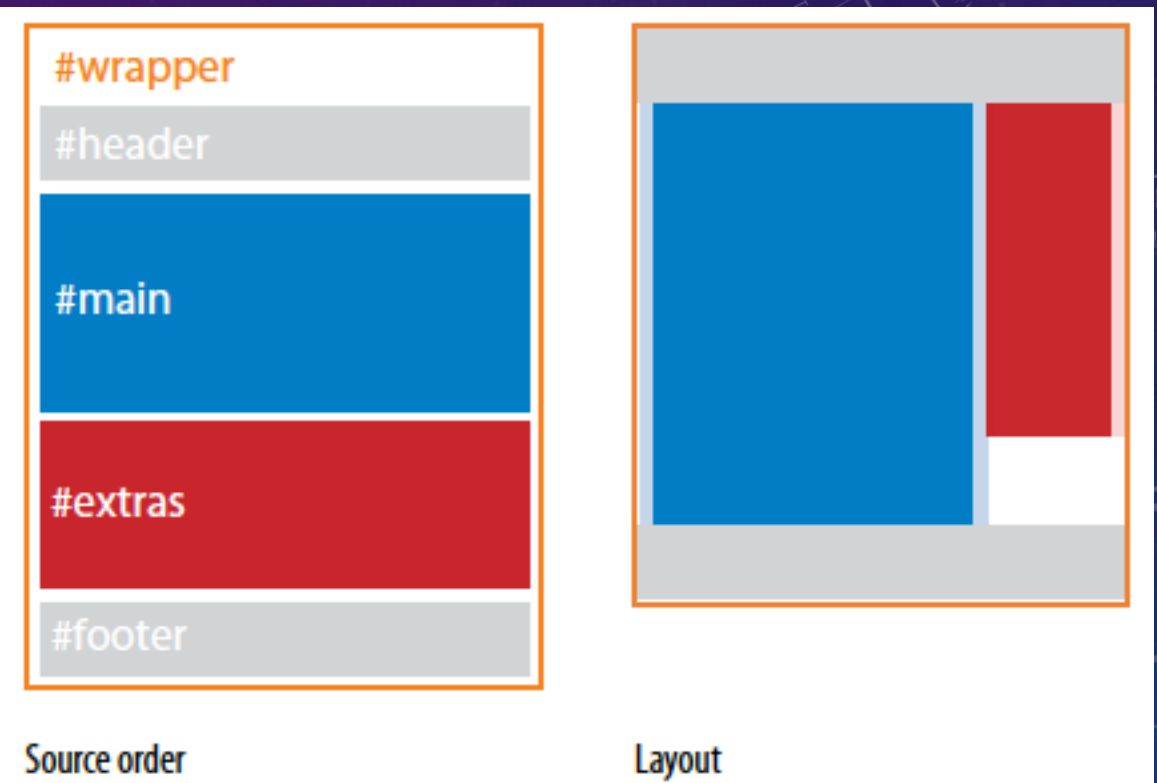
```
<div id="header">Masthead and headline</div>
<div id="main">Main article</div>
<div id="extras">List of links and news</div>
<div id="footer">Copyright information</div>
```

The styles

```
#main {
  float: left;
  width: 60%;
  margin: 0 5%;
}
#extras {
  float: left;
  width: 25%;
  margin: 0 5% 0 0;
}
#footer {
  clear: left;
}
```


FIXO 2 COLUNAS

- Div wrapper que envolve toda a página com `width` em px
- As divs contidas em wrapper são ajustadas com `float: left` e `width` em px
- Observar que a soma dos valores de `width` internas (main e extras) não excedam a div externa (wrapper)



FIXO 2 COLUNAS

- Adicione cores e conteúdo.
- Para centralizar o layout basta ajustar `margin-left` e `margin-right` para `auto`
- Como ficou a exibição?

The markup

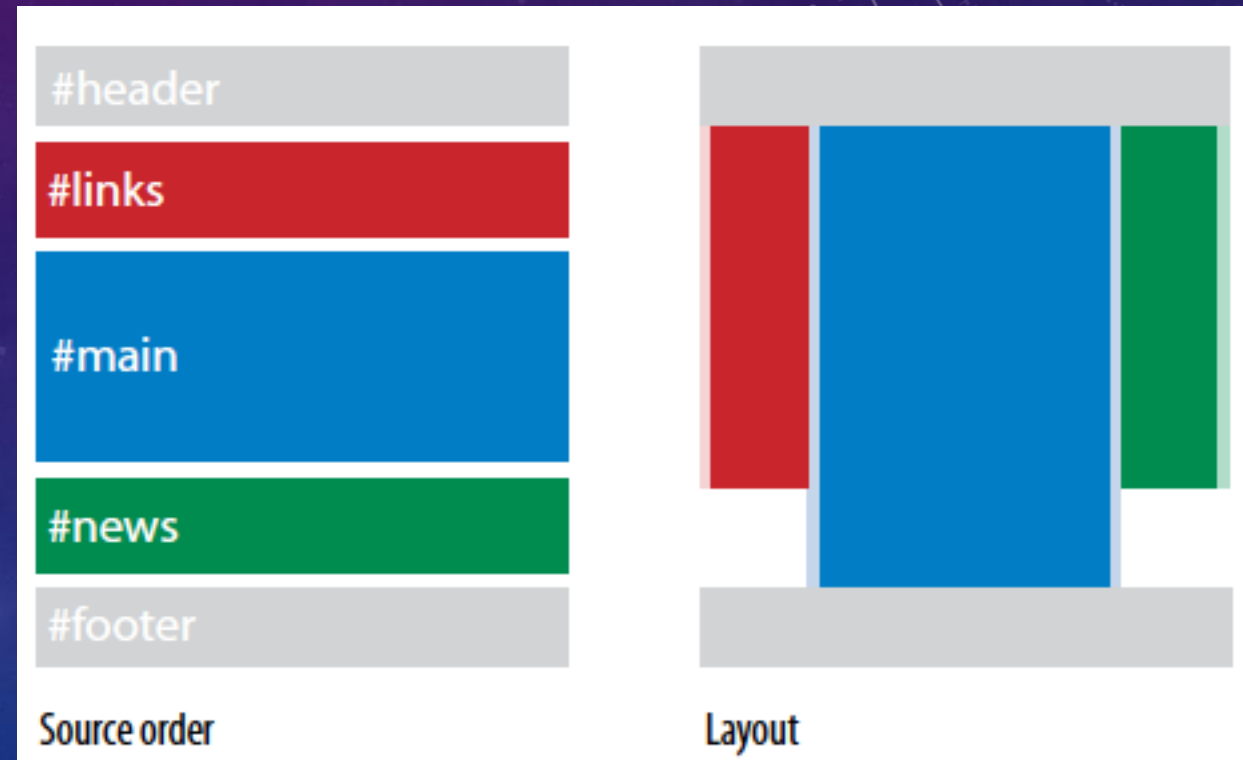
```
<div id="wrapper">  
  <div id="header">Masthead and headline</div>  
  <div id="main">Main article</div>  
  <div id="extras">List of links and news</div>  
  <div id="footer">Copyright information</div>  
</div>
```

The styles

```
#wrapper {  
  width: 960px;  
}  
#main {  
  float: left;  
  width: 650px;  
  margin: 0 20px;  
}  
#extras {  
  float: left;  
  width: 250px;  
  margin: 0 20px 0 0;  
}  
#footer {  
  clear: left;  
}
```

FLUID 3 COLUNAS

- Similar a estratégia de 2 colunas, exceto a margem extra na coluna central
- Ajustar width e float das 3 colunas
- A ordem dos elementos continua determinando a sequência de exibição
- Observar que a soma de margin e width de todos os elementos não deve passar de 100%



FLUID 3 COLUNAS

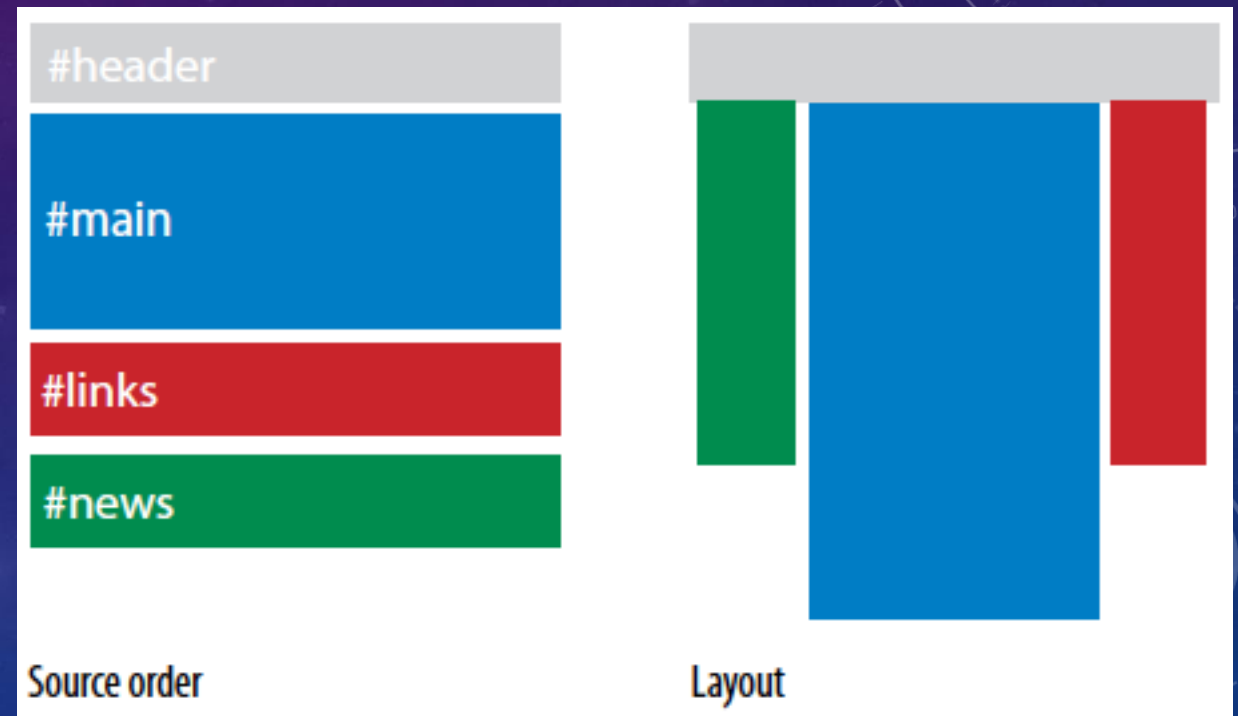
- Adicionar conteúdo e cores de fundo.
- Como ficou a exibição?

The styles

```
#links {  
  float: left;  
  width: 22.5%;  
  margin: 0 0 0 2.5%;  
}  
#main {  
  float: left;  
  width: 45%;  
  margin: 0 2.5%;  
}  
#news {  
  float: left;  
  width: 22.5%;  
  margin: 0 2.5% 0 0;  
}  
#footer {  
  clear: left;  
}
```


LAYOUTS POSICIONADOS

- Utilizam a propriedade `position` ao invés de `float` para criar as colunas
- Para layouts fluid de 3 colunas ajusta-se o `width` usando %



FLUID 3 COLUNAS

The markup

```
<div id="header">Masthead and headline</div>
<div id="content">
  <div id="main">Main article</div>
  <div id="news">News items</div>
  <div id="links">List of links</div>
</div>
```

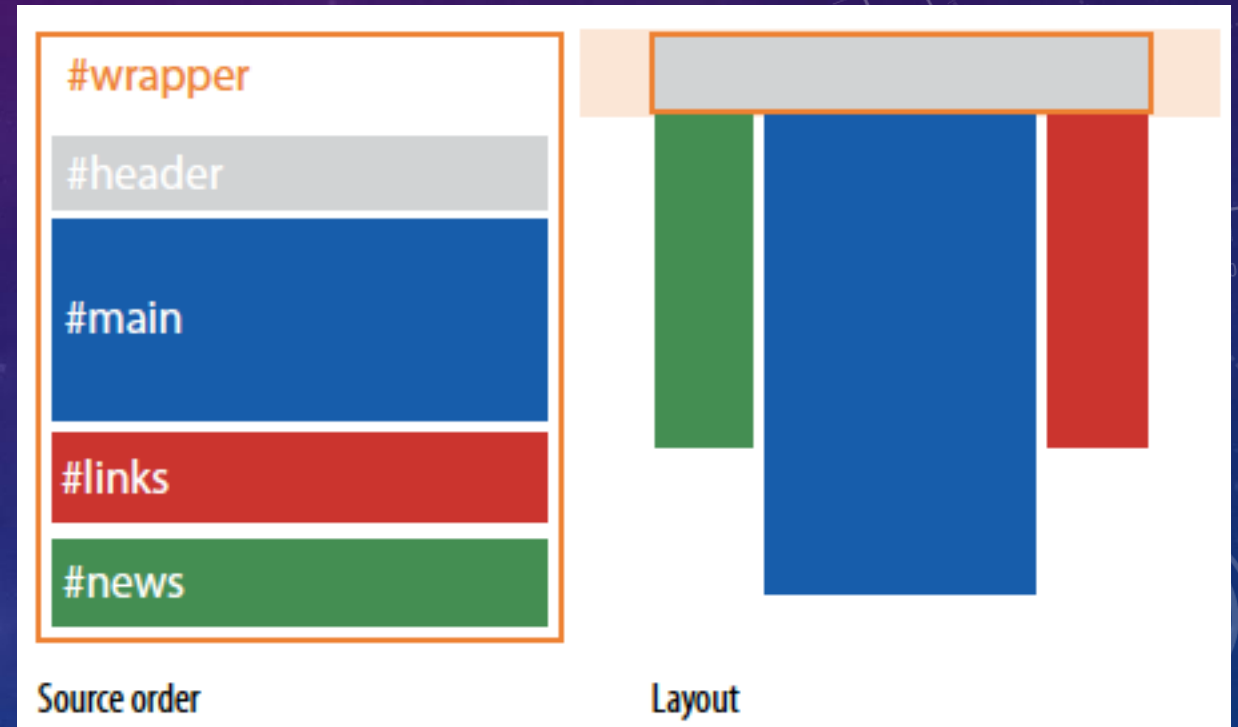
- A div header deve ficar sempre acima do restante, logo cria-se a div content para funcionar como containing box
 - Para isso, ajusta-se `position: relative`
- Todas as outras div têm `position: absolute` em relação a div content, que se tornou containing box
- Observar que margens e width não devem passar de 100%

The styles

```
#content {
  position: relative;
  margin: 0;
}
#main {
  width: 50%;
  position: absolute;
  top: 0;
  left: 25%;
  margin: 0;
}
#news {
  width: 20%;
  position: absolute;
  top: 0;
  left: 2.5%;
  margin: 0;
}
#links {
  width: 20%;
  position: absolute;
  top: 0;
  right: 2.5%;
  margin: 0;
}
```

FIXO 3 COLUNAS

- Largura das colunas definidas a nível de pixels
- A página completa é envolvida numa div (wrapper)



FIXO 3 COLUNAS

The styles

```
#wrapper {  
  width: 960px;  
  margin: 0 auto;  
}  
#content {  
  margin: 0;  
  position: relative;  
}  
#main {  
  width: 520px;  
  position: absolute;  
  top: 0;  
  left: 220px;  
  margin: 0;  
}
```

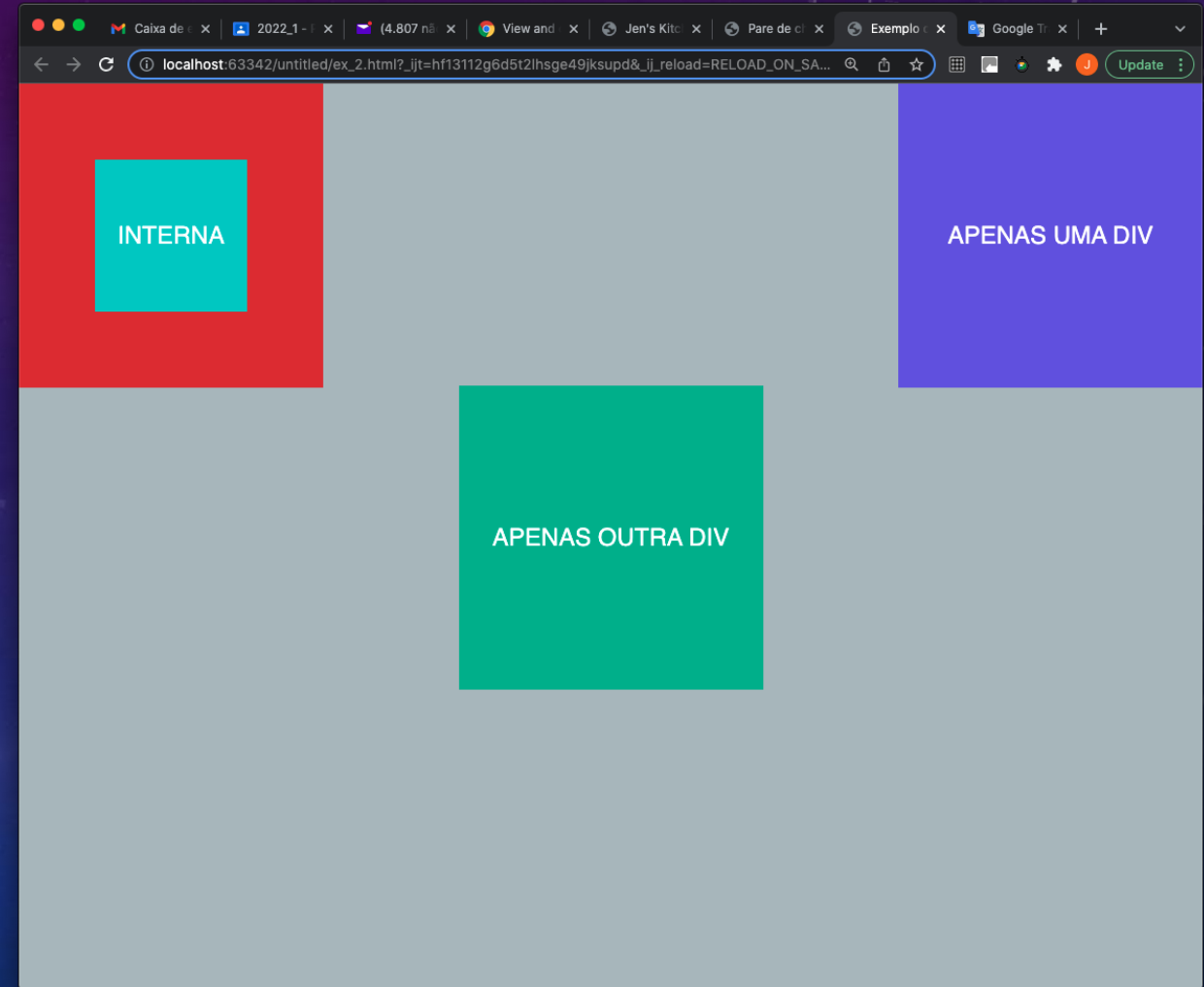
```
#news {  
  width: 200px;  
  position: absolute;  
  top: 0;  
  left: 0;  
  margin: 0;  
}  
#links {  
  width: 200px;  
  position: absolute;  
  top: 0;  
  right: 0;  
  margin: 0;  
}
```


EXERCÍCIO 1

- Implementar cada um dos layouts de exemplos mostrados anteriormente em uma página
 - a. Layout Fluid de 2 colunas*
 - b. Layout Fixo de 2 colunas*
 - c. Layout Fluid de 3 colunas*
 - d. Fluid de 3 colunas*
 - e. Fixo de 3 colunas*
- Acrescentar cores de fundo e uma quantidade de conteúdo que possa evidenciar as propriedades do layout

EXERCÍCIO 2

- Implementar as ações orientadas no tutorial do link abaixo
- O resultado final do tutorial é mostrado ao lado
- <https://medium.com/collabcode/pare-de-chutar-e-aprenda-como-funciona-a-position-absolute-ccd07c68b32b>



REFERÊNCIAS

- Learning Web Design, 4a. Ed
- <https://medium.com/collabcode/pare-de-chutar-e-aprenda-como-funciona-a-position-absolute-ccd07c68b32b>
- <https://alistapart.com/article/fluidgrids/>

